REMARKS

The Applicant and the undersigned thank Examiner Guill for his careful review of this application. The Applicant especially thanks Examiner Guill for his time and consideration given during the telephonic interview of January 13, 2006. A summary of this telephonic interview is provided below.

Upon entry of this amendment, Claims 1-10, 12, 15-16, and 24-25 have been cancelled and Claims 11, 23-14, 17-23, and 26-30 remain pending in this application. The independent claims are Claims 11, 20, and 29.

Consideration of the present application is respectfully requested in light of the above claim amendments to the application, the telephonic interview, and in view of the following remarks.

Summary of Telephonic Interview of January 13, 2006

The Applicant and the undersigned thank Examiner Guill for his time and consideration given during the telephonic interview of January 13, 2006. During this telephonic interview, a proposed amendment to the claims was discussed. The Applicant provided the proposed amendment to the claims in advance of the interview.

The Applicant's representative explained to Examiner Guill that the prior art of record does not provide any teaching of (1) automatically configuring a memory map of a virtual machine by (2) assigning areas of the memory map to receive predetermined types of data from the target program based on the file format to execute the target program; (3) the virtual machine being capable of executing the target program in one of three modes of operation: (4) a first mode of operation comprising a real mode (5) for executing programs comprising instructions based on DOS; (6) a second mode of operation for executing a target program comprising a high level programming language; and (7) a third mode of operation comprising a protected mode for executing a target program comprising thirty-two bit code. The Applicant's representative explained that the automatically configuring the memory map step is illustrated in Figure 4 of the original patent application.

It was explained that the prior art of record also does not (8) construct the virtual machine from one or more layered operating system shells that (9) correspond with the memory map so that the virtual machine is (10) capable of executing DOS and Windows type target programs;

and (11) simulate values of the computer system with the one or more layered operating system shells of the virtual machine. The operating system shells are supported by the original application on page 6 line 22 through page 7 line 2; page 9, lines 1-6; and page 10, line 15 through page 11, line 2. See also elements 201 of Figures 1 and 3 and element 211 of Figure 2.

After listening to the Applicant's representative, Examiner Guill provided a few suggestions to the claims in order to make them more clear under 35 U.S.C. §112, second paragraph. Specifically, Examiner Guill suggested that the Applicant amend the claims to provide steps which relate back to the problem or context set forth in the preamble of the claim. The Applicant's representative agreed to those suggestions and they have been adopted in this paper.

Examiner Guill also stated that he would review the proposed claims to determine if the terms such as "DOS" and "Windows" are permitted under the patent rules. The Examiner was not sure if these terms are generic. The Applicant's representative referred the Examiner to some pages of thc original detailed description in which Applicant's representative believed that the "DOS" and "Windows" terms were referenced with the intent that these terms are generic: for "DOS" see page 11, lines 4-5; for "Windows", see page 16, lines 19-22; page 21, lines 4-5.

Examiner Guill expressed that he understood the Applicant's representative comments and the concepts presented by the amended claims. Examiner Guill indicated that an update search would be conducted after the Applicant submits the claims in a formal amendment.

The Applicant and the undersigned request Examiner Guill to review this interview summary and to approve it by writing "Interview Record OK" along with his initials and the date next to this summary in the margin as discussed in MPEP § 713.04, p. 700-202.

Claim Rejections under 35 U.S.C. §112, first paragraph

The Examiner rejected Claim 27 under 35 U.S.C. §112, first paragraph as failing to comply with the written description requirement. The Examiner explains that Claim 27 recites loading a software CPU shell when the virtual machine operates in the first and second modes of operation. The Examiner notes that thc second mode of operation uses a high level language which does not load a software CPU shell. Compare Figure 2 that illustrates the second mode of operation with Figure 1 that illustrates the first mode of operation and Figure 3 that illustrates the third mode of operation.

The Applicant has amended the claims in light of the Examiner's helpful comments. The Applicant has amended Claim 27 to recite first and third modes of operation that support CPU shell loading instead of first and second modes of operation. Reconsideration and withdrawal of the Rejection under 35 U.S.C. § 112, first paragraph are respectfully requested.

Claim Rejections Under 35 U.S.C. § 103

The Examiner rejected Claims 11-16, 18-25, and 27-30 under 35 U.S.C. § 103(a) as being unpatentable over a 1995 article entitled "Dynamic Detection and Classification of Computer Viruses Using General Behavior Patterns," authored by Le Charlier et al. (hereinafter, the "Le Charlier reference") in view of a 1993 book entitled, "Inside Windows NT," authored by Helen Custer (hereinafter, the "Custer reference"), and further in view of U.S. Pat. No. 5,978,917 issued in the name of Chi et al. (hereinafter, the "Chi reference").

The Examiner rejected Claims 17 and 26 under 35 U.S.C. § 103(a) as being unpatentable over the Le Charlier, Custer, and Chi references, and further in view of U.S. Pat. No. 6,851,057 issued in the name of Nachenberg et al. (hereinafter, the "Nachenberg reference").

The Applicant respectfully offers remarks to traverse these pending rejections. The Applicant will address each independent claim separately as the Applicant believes that each independent claim is separately patentable over the prior art of record.

Independent Claim 11

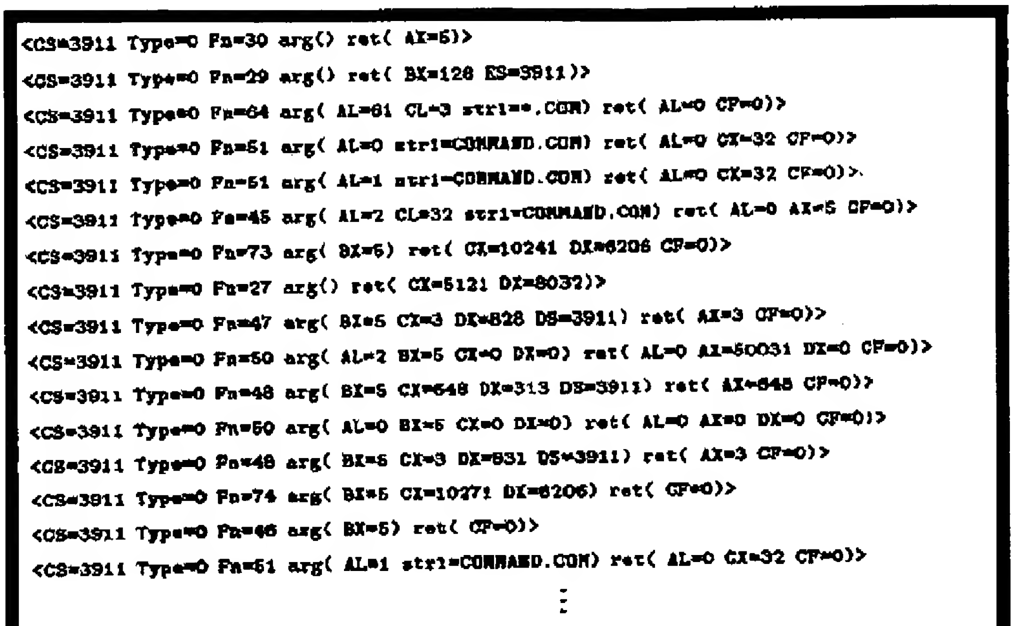
It is respectfully submitted that the Le Charlier, Custer, Chi, and Nachenberg references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) evaluating a file format of the target program; (2) evaluating control fields within a header of a file containing the target program; (3) automatically configuring a memory map of the virtual machine by (4) assigning areas of the memory map to receive predetermined types of data from the target program based on the file format in order to execute the target program, (5) the virtual machine being capable of executing the target program in one of three modes of operation based on the file format and the control fields within the header of the file, (6) a first mode of operation comprising a real mode (7) for executing programs comprising instructions based on DOS, (8) a second mode of operation for executing target programs comprising a high level programming language, and (9) a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code; (10) constructing the virtual machine from one or more layered operating system shells that correspond with the memory map so that (11) the virtual machine is capable of executing DOS and Windows type target programs; (12) simulating values of the computer system with the one or more layered operating system shells of the virtual machine; (13) setting and resetting behavior flags in a register in order to track behavior of the target program in response to the simulated values during execution of the target program by the virtual machine; (14) storing a sequence in which the behavior flags are set and reset in the register by the target program during execution of the target program by the virtual machine; (15) passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine; (16) terminating the virtual machine after execution of the target program, thereby removing from the computer system a copy of the target program that was contained within the virtual machine; and (17) evaluating the behavior flag data and sequence flag data with the computer system to determine if the target program contains malicious code, as recited in amended independent Claim 11.

The Le Charlier Reference

The Le Charlier reference describes how an emulator is used to monitor system activity of a virtual PC, and how an expert system ASAX is used to analyze the stream of data that the emulator produced.

The Le Charlier reference also describes how general rules are used to generally detect real viruses reliably, and specific rules to extract details of their behavior. See the Le Charlier reference, page 1.

Sections 4.5 and 4.6 of the Le Charlier reference describe an audit trail of an emulator. The audit trail includes parameters popped from a stack with return values. See Figure 3 of the Le Charlier reference reproduced below and page 15.



```
<CS=3911 Type=0 Fn=30 arg() ret( AX=5)>
<CS=3911 Type=0 Fn=29 arg() ret( BX=128 ES=3911)>
<CS=3911 Type=0 Fn=64 arg( AL=61 CL=3 str1=*.COM) ret( AL=0 CF=0)>
<CS=3911 Type=0 Fn=51 arg( AL=0 str1=COMMAND.COM) ret( AL=0 CX=32 CF=0)>
<CS=3911 Type=0 Fn=61 arg( AL=1 str1=COMMAND.COM) ret( AL=0 CX=32 CF=0)>
<CS=3911 Type=0 Fn=45 arg( AL=2 CL=32 str1=COMMAND.COM) ret( AL=0 AX=5 CF=0)>
<CS=3911 Type=0 Fn=73 arg( BX=5) ret( CX=10241 DX=6206 CF=0)>
<CS=3911 Type=0 Fn=27 arg() ret( CX=5121 DX=8032)>
<CS=3911 Type=0 Fn=47 arg( BX=5 CX=3 DX=828 DS=3911) ret( AX=3 CF=0)>
<CS=3911 Type=0 Fn=50 arg( AL=2 BX=5 CX=0 DX=0) ret( AL=0 AX=50031 DX=0 CF=0)>
<CS=3911 Type=0 Fn=48 arg( BX=5 CX=648 DX=313 DS=3911) ret( AX=648 CF=0)>
<CS=3911 Type=0 Fn=50 arg( AL=0 BX=5 CX=0 DX=0) ret( AL=0 AX=0 DX=0 CF=0)>
<CS=3911 Type=0 Fn=48 arg( BX=5 CX=3 DX=831 DS=3911) ret( AX=3 CF=0)>
<CS=3911 Type=0 Fn=74 arg( BX=5 CX=10271 DX=6206) ret( CF=0)>
<CS=3911 Type=0 Fn=46 arg( BX=5) ret( CF=0)>
<CS=3911 Type=0 Fn=61 arg( AL=1 str1=COMMAND.COM) ret( AL=0 CX=32 CF=0)>
                                    ⋮
```

Figure 3: Excerpt from an audit trail for the Vienna virus

Figure 3 of the Le Charlier reference is an example of an audit trail. The audit trail of this figure is a human representation of a binary NADF file. The example illustrated is from the Vienna virus.

The Examiner admits that the Le Charlier reference fails to provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, as recited in amended independent Claim 11.

Further, the Le Charlier reference is only designed to execute DOS type target programs and not Windows type target programs. The Le Charlier reference does not automatically configure a memory map of a virtual machine by assigning areas of the memory map to receive predetermined types of data from the target program based on the file format in order to execute the target program. The Le Charlier reference also does not construct a virtual machine from one or more layered operating system shells that correspond with the memory map so that the virtual machine is capable of executing DOS and Windows type target programs. The Le Charlier reference does not simulate values of the computer system with the one or more layered operating system shells of the virtual machine, as recited in amended independent Claim 11.

The Custer Reference

The Examiner admits that the Le Charlier reference does not provide any teaching of a software processor that executes 32-bit or 64-bit code. The Examiner also admits that the Le Charlier reference also does not provide any teaching of an operating system simulation shell that responds to application program interface calls. To make up for the 32-bit code, operating system shell, and evaluating the file format deficiencies, the Examiner relies upon the Custer reference.

The Examiner explains that since the Custer reference on page 150, second paragraph, describes a MS-DOS operating system being simulated, then it is obvious that the operating simulation shell responds to application program interface calls. However, the Custer reference does not describe the virtual machine context of the present invention for detecting malicious computer code.

Specifically, the Custer reference, being a general technical document, does not provide any teaching of evaluating a file format of the target program; evaluating control fields within a header of a file containing the target program; automatically configuring the virtual machine to execute the target program in one of three modes of operation based on the file format and the control fields within the header of the file, a first mode of operation comprising a real mode, a second mode of operation for executing target programs comprising a high level programming language, a third mode of operation comprising a protected mode for executing target programs comprising thirty-two bit code, and passing behavior flag data and sequence flag data from the virtual machine to the computer system for evaluation after execution of the target program by the virtual machine, as recited in amended independent Claim 11.

Further, the Custer reference also fails to execute both DOS type and Windows type target programs. Like the Le Charlier reference, the Custer reference does not automatically configure a memory map of a virtual machine by assigning areas of the memory map to receive predetermined types of data from the target program based on the file format in order to execute the target program. The Custer reference, like the Le Charlier reference, also does not construct a virtual machine from one or more layered operating system shells that correspond with the memory map so that the virtual machine is capable of executing DOS and Windows type target programs. The Custer reference does not simulate values of the computer system with the one or more layered operating system shells of the virtual machine and Custer does not provide any teaching of evaluating the behavior flag data and sequence flag data with the computer system to determine if the target program contains malicious code, as recited in amended independent Claim 11.

<u>The Chi Reference</u>

The Examiner admits that the Le Charlier reference does not provide any teaching of a second mode of operation for executing target programs that comprise a high level programming language. To address this deficiency of the Le Charlier reference, the Examiner relies upon the Chi reference.

The Chi reference describes detecting the presence of macro viruses within a digital computer (1). An application program (5) that is associated with the digital computer (1) generates at least one local document (11). Macros contained within the global environment (13) and the local document(s) (11) are executed in a simulated manner by an emulator (15). A preselected decision criterion is used by a detection module (17) to determine when a macro virus is present. See Figure 4 of the Chi reference reproduced below and column 1, lines 46-56.

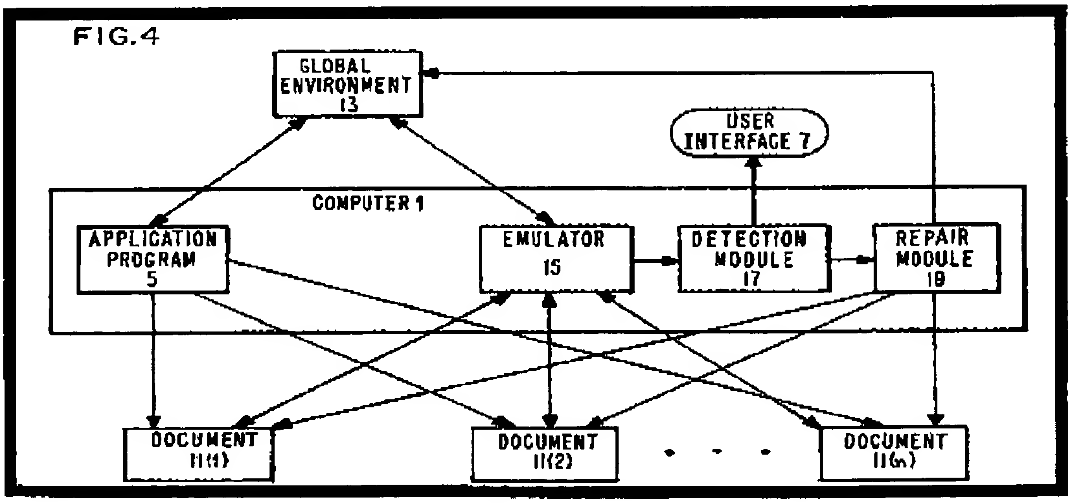


FIG.4

However, like the Le Charlier reference, the Chi reference does not automatically configure a memory map of a virtual machine by assigning areas of the memory map to receive predetermined types of data from the target program based on the file format in order to execute the target program. The Chi reference, like the Le Charlier reference, also does not construct a virtual machine from one or more layered operating system shells that correspond with the memory map so that the virtual machine is capable of executing DOS and Windows type target programs. The Chi reference does not simulate values of the computer system with the one or more layered operating system shells of the virtual machine and the Chi reference does not provide any teaching of evaluating the behavior flag data and sequence flag data with the computer system to determine if the target program contains malicious code, as recited in amended independent Claim 11.

The Nachenberg Reference

The Examiner admits that the Le Charlier reference does not provide any teaching of a virtual machine that executes a target program at each entry point defined within an entry point table. To address this deficiency of the Le Charlier referenc, the Examiner refers the Applicant to the Nachenberg reference.

The Nachenberg reference describes a virus detection system that uses virus signatures and partial emulation of target code. The Nachenberg reference does not use a complete virtual machine, but instead, a virtual machine that executes portions of code.

Specifically, the Nachenberg reference describes an engine 414 that contains an emulating module 426 for emulating code in the file 100 starting at an entry point. The emulating module 426 includes a control program 428 for setting up a virtual machine 430 having a virtual processor 432 and an associated virtual memory 434. The virtual machine can emulate a 32-bit MICROSOFT WINDOWS environment, an APPLE MACINTOSH environment, or any other environment for which emulation is desired. See Figure 4 of the Nachenberg reference reproduced below.

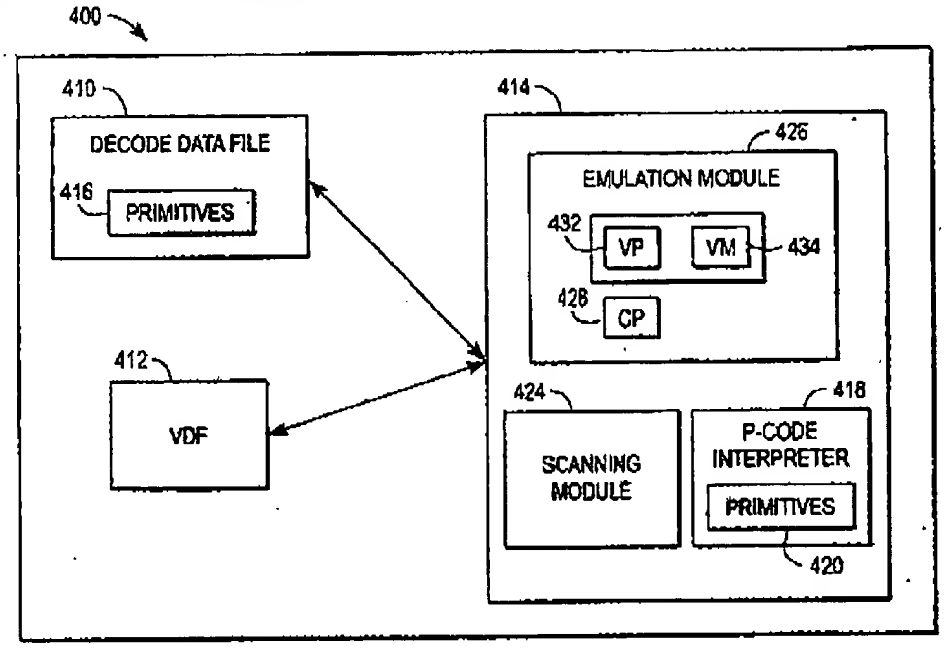


FIG. 4

The virtual machine of the Nachenberg reference uses the virtual processor 432 to execute selected code in the virtual memory 434 in isolation from the remainder of the computer system. Emulation as taught by the Nachenberg reference starts with a given context, which specifies the contents of the registers, stacks, etc. in the virtual processor 432. During emulation, every page of virtual memory 434 that is read from, written to, or emulated through is marked. The number of instructions that the virtual machine 430 emulates can be fixed at the beginning of emulation or can be determined adaptively while the emulation occurs. See Nachenberg reference, column 8, lines 6-20. With the virtual machine 430 only running a finite number of instructions, it is evident to one of ordinary skill in the art that this means that the system of Nachenberg is not a complete virtual machine: the Nachenberg system does not completely execute an entire target program – it does not set and reset behavior flags "during execution of the target program by the virtual machine," as recited in amended independent Claim 11.

Further, like the Le Charlier reference, the Nachenberg reference does not construct a virtual machine from one or more layered operating system shells that correspond with a memory map so that the virtual machine is capable of executing DOS and Windows type target programs. The Nachenberg reference does not simulate values of the computer system with the one or more layered operating system shells of the virtual machine for execution of the target program. The Nachenberg reference does not provide any teaching of evaluating the behavior flag data and sequence flag data with the computer system to determine if the target program contains malicious code, as recited in amended independent Claim 11.

Consider Claimed Combination as a Whole

The Applicant reminds the Examiner that the M.P.E.P. is clear on the subject of considering the differences between the claims and the prior art. The Examiner must consider the claimed combination as a whole as taught by M.P.E.P. § 2141.02 (Rev. 3, August 2005) that states the following:

> "I. THE CLAIMED INVENTION AS A WHOLE MUST BE CONSIDERED
>
> In determining the differences between the prior art and the claims, the question under 35 U.S.C. § 103 is not whether the differences themselves would have been obvious, but whether the claimed invention as a whole would have been obvious. Stratoflex, Inc. v. Aeroquip Corp., 713 F.2d 1530, 218 USPQ 871 (Fed. Cir. 1983)." [Emphasis Supplied.]

The Applicant respectfully submits that the Examiner is over looking the specific design of Applicant's claimed virtual machine invention for detecting malicious code and the generic designs presented by the prior art references that do not mention or even suggest virtual machines for detecting malicious computer code. For example, the Custer reference is only a generic textbook that discusses the operations of the WINDOWS NT operating system. To address the virtual characteristics and the design of the Applicant's claimed technology, the Examiner relies upon the Custer reference in combination with the other references. The Applicant submits that the Examiner is not considering the claimed combination of elements as a whole, but is instead, he is looking at only the differences between the claimed combination and what elements exist in the prior art.

Conclusion Regarding Independent Claim 11

In light of the differences between Claim 11 and the Le Charlier, Custer, Chi, and Nachenberg references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in new independent Claim 11. Accordingly, consideration and allowance of new independent Claim 11 are respectfully requested.

Independent Claim 20

It is respectfully submitted that the Le Charlier, Custer, Chi, and Nachenberg references references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) a processing unit; (2) a memory storage device; and (3) one or more program modules stored in said memory storage device for providing instructions to said processing unit; (4) the processing unit executing said instructions of said one or more program modules, operable for (5) evaluating a file format of the target program; (6) evaluating control fields within a header of a file containing the target program; (7) automatically configuring a memory map of a virtual machine (8) by assigning areas of the memory map to receive predetermined types of data from the target program based on the file format in order to execute the target program, (9) the virtual machine being capable of executing the target program (10) in one of three modes of operation (11) based on the file format and the control fields within the header of the file, (12) a first mode of operation comprising a real mode for executing programs comprising instructions based on DOS, (13) a second mode of operation for executing target programs comprising a high level programming language, and (14) third mode of operation for executing target programs comprising thirty-two bit code; (15) constructing the virtual machine from one or more layered operating system shells that correspond with the memory map so that the virtual machine is capable of executing DOS and Windows type target programs; (16) simulating values of the computer system with the one or more layered operating system shells of the virtual machine; (17) setting and resetting behavior flags in a register in order (18) to track behavior of the target program in response to the simulated values during execution of the target program by the virtual machine; (18) storing a sequence in which the behavior flags are set and reset in the register by the target program during execution of the target program by the virtual machine; (19) passing behavior flag data and sequence flag data from the virtual machine to the computer system after execution of the target program by the virtual machine; and (20) evaluating the behavior flag data and sequence flag data with the computer system (21) to determine if the target program contains malicious code, as recited in amended independent Claim 20.

Similar to Claim 1, Claim 20 describes constructing the virtual machine from one or more layered operating system shells that correspond with the memory map so that the virtual machine is capable of executing DOS and Windows type target programs. As noted above, none of the prior art references relied upon by the Examiner teach this dual-function capability.

In light of the differences between Claim 20 and the Le Charlier, Custer, Chi, and Nachenberg references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 20. Accordingly, consideration and allowance of amended independent Claim 20 are respectfully requested.

Independent Claim 29

It is respectfully submitted that the Le Charlier, Custer, Chi, and Nachenberg references references, individually or in view of each other, fail to describe, teach, or suggest the combination of: (1) automatically configuring a memory map of a virtual machine (2) by assigning areas of the memory map to receive predetermined types of data from the target program based on the file format to execute the target program, (3) the virtual machine being capable of executing the target program (4) in one of three modes of operation, (5) a first mode of operation comprising a real mode for executing programs comprising instructions based on DOS, (6) a second mode of operation for executing a target program comprising a high level programming language, and (7) a third mode of operation comprising a protected mode for executing a target program comprising thirty-two bit code; (8) constructing the virtual machine from one or more layered operating system shells that correspond with the memory map so that the virtual machine is capable of executing DOS and Windows type target programs; (9) simulating values of the computer system with the one or more layered operating system shells of the virtual machine; (10) setting and resetting behavior flags in a register in order to track behavior of the target program in response to the simulated values during execution of the target program by the virtual machine; (11) storing a sequence in which the behavior flags are set and reset in the register by the target program during execution of the target program by the virtual machine; (12) passing behavior flag data and sequence flag data from the virtual machine to a computer system after execution of the target program by the virtual machine; (13) terminating the virtual machine after execution of the target program, thereby removing from the computer system a copy of the target program that was contained within the virtual machine; and (14) evaluating the behavior flag data and sequence flag data with the computer system to determine if the target program contains malicious code, as recited in amended independent Claim 29.

Similar to Claim 1, Claim 29 describes constructing the virtual machine from one or more layered operating system shells that correspond with the memory map so that the virtual machine is capable of executing DOS and Windows type target programs. As noted above, none of the prior art references relied upon by the Examiner teach this dual-function capability.

In light of the differences between Claim 29 and the Le Charlier, Custer, Chi, and Nachenberg references mentioned above, one of ordinary skill in the art recognizes that the combination proposed by the Examiner cannot anticipate or render obvious the recitations as set forth in amended independent Claim 29. Accordingly, consideration and allowance of amended independent Claim 29 are respectfully requested.

Dependent Claims 13-14, 17-19, 21-23, 26-28 and 30

The Applicant respectfully submits that the above-identified dependent claims are allowable because the independent claims from which they depend are patentable over the cited references. The Applicant also respectfully submits that the recitations of these dependent claims are of patentable significance.

In view of the foregoing, the Applicant respectfully requests that the Examiner withdraw the pending rejections of dependent Claims 13-14, 17-19, 21-23, 26-28 and 30.

CONCLUSION

The foregoing is submitted as a full and complete response to the Office Action mailed on July 20, 2005. The Applicant and the undersigned thank Examiner Guill for consideration of these remarks. The Applicant has amended the claims to overcome the prior art. The Applicant respectfully submits that the present application is in condition for allowance. Such action is hereby courteously solicited.

If the Examiner believes that there are any issues that can be resolved by a telephone conference, or that there are any formalities that can be corrected by an Examiner's amendment, please contact the undersigned in the Atlanta Metropolitan area (404) 572-2884.

Respectfully submitted,

Steven P. Wigmore
Reg. No. 40,447
January 20, 2006

King & Spalding LLP
191 Peachtree Street, N.E.
Atlanta, Georgia 30303-1763
telephone: (404) 572.4600
K&S File No. 05456-105039